Package 'gctsc'

December 17, 2025

Title Modeling Count Time Series Data via Gaussian Copula Models

Version 0.1.3

Description Gaussian copula models for count time series. Includes simulation utilities, likelihood approximation, maximum-likelihood estimation, residual diagnostics, and predictive inference. Implements the Time Series Minimax Exponential Tilting (TMET) method, an adaptation of Minimax Exponential Tilting (Botev, 2017) <doi:10.1111/rssb.12162> and the Vecchia-based tilting framework of Cao and Katzfuss (2025) <doi:10.1080/01621459.2025.2546586>. Also provides a linear-cost implementation of the Geweke–Hajivassiliou–Keane (GHK) simulator inspired by Masarotto and Varin (2012) <doi:10.1214/12-EJS721>, and the Continuous Extension (CE) approximation of Nguyen and De Oliveira (2025) <doi:10.1080/02664763.2025.2498502>. The package follows the S3 structure of 'gcmr', but all code in 'gctsc' was developed independently.

License MIT + file LICENSE

Imports Rcpp, Matrix, TruncatedNormal, VGAM, car, truncnorm

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/QNNHU/gctsc

BugReports https://github.com/QNNHU/gctsc/issues

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, VeccTMVN, gcmr, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 3.5)

LazyData false

NeedsCompilation yes

Author Quynh Nguyen [aut, cre],
Victor De Oliveira [aut]

Maintainer Quynh Nguyen <nqnhu2209@gmail.com>

Repository CRAN

Date/Publication 2025-12-17 10:20:26 UTC

Contents

`arma.cormat` *ARMA Correlation Structure for Copula Time Series*

Description

Constructs a correlation structure object for use in Gaussian copula time series models with autoregressive moving average (ARMA) dependence.

Usage

```
arma.cormat(p = 0, q = 0, tau.lower = NULL, tau.upper = NULL)
```

Arguments

`p`	Integer. AR order (non-negative).
`q`	Integer. MA order (non-negative).
`tau.lower`	Optional vector of lower bounds for the ARMA parameters.
`tau.upper`	Optional vector of upper bounds for the ARMA parameters.

Value

An object of class `"arma.gctsc"` and `"cormat.gctsc"`, containing:

`npar`	Number of ARMA parameters.
`od`	A length-2 vector `c(p, q)` giving the AR and MA order.
`start`	Function to compute starting values from data using `arima`.

See Also

gctsc, poisson.marg, predict.gctsc

---

campyl *Weekly Campylobacter case counts across Germany*

Description

Weekly Campylobacter case counts across Germany

Usage

```
data("campyl")
```

Format

A data frame with 1248 rows and 413 variables.

---

coef.gctsc *Extract Coefficients from a gctsc Model*

Description

Returns the estimated coefficients from a fitted gctsc model object.

Usage

```
## S3 method for class 'gctsc'
coef(object, ...)
```

Arguments

object	An object of class gctsc.
...	Ignored. Included for S3 method compatibility.

Value

A named numeric vector containing the estimated model coefficients, including both marginal parameters and copula dependence parameters.

gctsc *Fit a Gaussian Copula Time Series Model for Count Data*

Description

Fits a Gaussian copula model to univariate count time series using discrete marginals (Poisson, negative binomial, binomial/beta–binomial, and zero–inflated variants) and latent dependence through ARMA correlation structures. The multivariate normal rectangle probability is evaluated using Minimax Exponential Tilting (TMET), the Geweke–Hajivassiliou–Keane (GHK) simulator, or the Continuous Extension (CE) approximation.

Usage

```
gctsc(
  formula = NULL,
  data,
  marginal,
  cormat,
  method = c("TMET", "GHK", "CE"),
  c = 0.5,
  QMC = TRUE,
  pm = 30,
  start = NULL,
  options = gctsc.opts()
)
```

Arguments

formula	A formula (e.g., y ~ x1 + x2) or, for zero–inflated marginals, a named list of formulas list(mu = ..., pi0 = ...).
data	A data frame containing y and covariates referenced in the formula(s).
marginal	A marginal model object such as poisson.marg, negbin.marg, zib.marg, or zibb.marg. See marginal.gctsc for the full list of supported marginals.
cormat	A correlation structure such as arma.cormat.
method	One of "TMET", "GHK", or "CE".
c	Smoothing constant for the CE approximation (ignored for TMET/GHK). Default is 0.5.
QMC	Logical; use quasi–Monte Carlo sampling for simulation–based likelihood approximations.
pm	Integer; truncated AR order used to approximate ARMA(p, q) when $q > 0$ (TMET only).
start	Optional numeric vector of starting values (marginal parameters followed by dependence parameters).

`options` Optional list specifying the random seed and Monte Carlo sample size for the likelihood approximation. If omitted, default values from `gctsc.opts()` are used (i.e., `seed = NULL` and `M = 1000`).

For simulation–based likelihoods (`method = "GHK"` or `method = "TMET"`), providing a fixed `seed` (e.g., `options = gctsc.opts(seed = 123)`) is **strongly recommended** to ensure reproducible and numerically stable likelihood evaluations. Without a fixed seed, the approximate log-likelihood may vary slightly between evaluations, which can affect numerical optimization.

See `gctsc.opts` for additional tuning parameters and their default values.

Details

The interface mirrors `glm()`. Zero–inflated marginals accept a list of formulas, for example `list(mu = y ~ x, pi0 = ~ z)`. Non–zero–inflated marginals accept a single formula (e.g., `y ~ x1 + x2`).

Formula handling For zero–inflated marginals:

- If neither `mu` nor `pi0` is supplied, both default to intercept-only models (`mu ~ 1`, `pi0 ~ 1`).
- If `mu` is supplied but `pi0` is omitted, `pi0 ~ 1` is used.

Dependence structure ARMA parameters are determined by `cormat`. ARMA(0,0) is not supported.

Method-specific notes

- CE ignores `QMC` and `options$M`.
- TMET and GHK require `options$M` to be a positive integer.
- TMET optionally uses `pm` when $q > 0$.

Value

An object of class `"gctsc"` containing:

`coef` Numeric vector containing the parameter estimates, with marginal parameters first and dependence parameters last.

`maximum` The maximized approximate log-likelihood value, returned on the internal minus scale used by the optimizer.

`hessian` Estimated Hessian matrix at the optimum, when available.

`se` Standard errors from the inverse Hessian, or NA if unavailable or not positive-definite.

`marginal` The marginal model object used (e.g., Poisson, negative binomial, ZIB, ZIBB).

`cormat` The correlation structure object used, typically created by `arma.cormat()`.

`ibeta` Integer vector giving the indices of the marginal parameters within `coef`.

`itau` Integer vector giving the indices of the dependence parameters within `coef`.

`nbeta` Number of marginal parameters.

`ntau` Number of dependence parameters.

`options` List of fitting and optimization controls, typically created by `gctsc.opts()`.

call	The matched call.
formula	The model formula(s) used to specify the marginal mean and zero-inflation components.
terms	Terms objects for the marginal model(s).
model	The model frame used for estimation (returned only when needed).
x	Design matrix for the marginal mean component.
z	Design matrix for the zero-inflation component (ZIB or ZIBB only), or NULL otherwise.
y	The response vector.
n	Number of observations used in the fit.
method	Character string identifying the likelihood approximation used (TMET, GHK, or CE).
QMC	Logical flag indicating whether quasi-Monte Carlo sampling was used.
pm	Truncated AR order used to approximate ARMA(p,q) when $q > 0$ (TMET only).
convergence	Optimizer convergence code (0 indicates successful convergence).

The returned object can be used with summary.gctsc, predict.gctsc, residuals.gctsc, and plot.gctsc.

See Also

arma.cormat, poisson.marg, zib.marg, zibb.marg, gctsc.opts

Examples

```
set.seed(42)
n <- 200
y <- sim_poisson(mu = 10, tau = 0.3, arma_order = c(1,0), nsim = n)$y
fit <- gctsc(y ~ 1,
  marginal = poisson.marg(lambda.lower = 0),
  cormat = arma.cormat(p = 1, q = 0),
  method = "CE",
  options = gctsc.opts(M = 1000, seed = 42))
summary(fit)
```

---

gctsc.opts *Set Options for Gaussian Copula Time Series Model*

Description

Creates a control list for simulation and likelihood approximation in the Gaussian copula model, including the random seed and Monte Carlo settings.

Usage

```
gctsc.opts(seed = NULL, M = 1000, ...)
```

Arguments

seed	Integer. Random seed for reproducibility (default: a random integer between 1 and 100000).
M	Integer. Number of Monte Carlo samples used in the likelihood approximation (default: 1000).
...	Ignored. Included for S3 method compatibility.

Value

A list with components:

seed	Integer. The random seed used.
M	Integer. Number of Monte Carlo samples.
opt	A function used internally by `gctsc()` to perform optimization of the approximate log-likelihood.

KCWC *Daily aggregated weather measurements for KCWC station*

Description

Daily aggregated weather measurements for KCWC station

Usage

```
data("KCWC")
```

Format

A data frame with 2665 rows and 4 variables.

marginal.gctsc *Marginal Models for Copula Time Series*

Description

The following marginal models are currently available:

`poisson.marg(link = "log")` Poisson distribution.

`negbin.marg(link = "log")` Negative binomial distribution.

`binom.marg(link = "logit", size)` Binomial distribution with fixed number of trials.

`bbinom.marg(link = "logit", size)` Beta-binomial with overdispersion.

`zip.marg(link = "log")` Zero-inflated Poisson model.

`zib.marg(link = "logit", size)` Zero-inflated binomial.

`zibb.marg(link = "logit", size)` Zero-inflated beta-binomial with separate covariates for zero inflation.

Usage

```
poisson.marg(link = "identity", lambda.lower = NULL, lambda.upper = NULL)

binom.marg(link = "logit", size = NULL, lambda.lower = NULL, lambda.upper = NULL)

zib.marg(link = "logit", size = NULL, lambda.lower = NULL, lambda.upper = NULL)

negbin.marg(link = "identity", lambda.lower = NULL, lambda.upper = NULL)

zip.marg(link = "identity", lambda.lower = NULL, lambda.upper = NULL)

bbinom.marg(link = "logit", size, lambda.lower = NULL, lambda.upper = NULL)

zibb.marg(link = "logit", size,  lambda.lower = NULL, lambda.upper = NULL)
```

Arguments

`link`	The link function for the mean (e.g., `"log"`, `"logit"`, `"identity"`).
`lambda.lower`	Optional lower bounds on parameters.
`lambda.upper`	Optional upper bounds on parameters.
`size`	Number of trials (for binomial-type models).

Details

These functions define the marginal distributions used in copula-based count time series models.

Each marginal constructor returns an object of class `"marginal.gctsc"` which defines:

- `start`: a function to compute starting values.

- npar: number of parameters.
- bounds: truncation bounds on the latent Gaussian.

These marginals are designed to work with gctsc() and its related methods.

Value

A list of class "marginal.gctsc" representing the marginal model.

References

Cribari-Neto, F. and Zeileis, A. (2010). Beta regression in R. *Journal of Statistical Software*, **34**(2): 1–24.

Ferrari, S.L.P. and Cribari-Neto, F. (2004). Beta regression for modeling rates and proportions. *Journal of Applied Statistics*, **31**(7): 799–815.

Masarotto, G. and Varin, C. (2012). Gaussian copula marginal regression. *Electronic Journal of Statistics*, **6**: 1517–1549.

See Also

gctsc, predict.gctsc, arma.cormat

Examples

```
poisson.marg(link = "identity")
zibb.marg(link = "logit", size = 24)
```

---

plot.gctsc *Diagnostic Plots for Fitted Gaussian Copula Time Series Models*

Description

Produces a set of diagnostic plots based on randomized quantile residuals and PIT values for objects of class gctsc.

Usage

```
## S3 method for class 'gctsc'
plot(
  x,
  caption = rep("", 5),
  main = rep("", 5),
  level = 0.95,
  col.lines = "gray",
  ...
)
```

Arguments

x	An object of class gctsc, the result of a call to gctsc.
caption	Optional character vector of length 6 to use as captions for each plot.
main	Optional main title for the plots (recycled if shorter than the number of plots shown).
level	Confidence level for the Q–Q plot envelope (default is 0.95).
col.lines	Color for reference lines in residual and ACF/PACF plots.
...	Additional arguments passed to plotting functions.

Details

The function displays up to five plots: time series of residuals, Q–Q plot, PIT histogram, and ACF/PACF plots of the residuals. These plots help assess model fit and potential misspecification.

The five diagnostic plots shown are:

1. Time series plot of randomized quantile residuals.
2. Q–Q plot comparing residuals to a standard normal distribution.
3. Histogram of probability integral transform (PIT) values.
4. Autocorrelation function (ACF) of the residuals.
5. Partial autocorrelation function (PACF) of the residuals.

Value

This function is called for its side effects and returns invisible().

See Also

residuals.gctsc for computing the residuals used in the plots.

Examples

```
# Simulate data from a Poisson AR(1) model
set.seed(123)
n <- 2000
mu <- 5
phi <- 0.5
arma_order <- c(1, 0)
y <- sim_poisson(mu = mu, tau = phi, arma_order = arma_order, nsim = n)$y

# Fit the model using the CE method
fit <- gctsc(y~1,
  marginal = poisson.marg(link = "identity", lambda.lower = 0),
  cormat = arma.cormat(p = 1, q = 0),
  method = "CE",
  options = gctsc.opts(seed = 1, M = 1000),
  c = 0.5
)
```

```
# Produce diagnostic plots
plot(fit)
```

---

pmvn_ce *Approximate Log-Likelihood via Continuous Extension (CE)*

Description

Computes the approximate log-likelihood of a count time series model using the Continuous Extension (CE) method. This approach approximates the probability of the observed count vector by replacing the discrete indicator function with a smooth approximation, controlled by a smoothing parameter c.

Usage

```
pmvn_ce(lower, upper, tau, od, c = 0.5, ret_llk = TRUE)
```

Arguments

lower	Numeric vector of length n; lower bounds of the transformed latent variables.
upper	Numeric vector of length n; upper bounds of the transformed latent variables.
tau	Numeric vector of ARMA dependence parameters (concatenated phi, theta).
od	Integer vector of length 2: c(p, q) for AR and MA orders.
c	Smoothing bandwidth parameter; default is 0.5. Must lie in (0,1).
ret_llk	Logical; return log-likelihood if TRUE.

Details

The method is applicable to Gaussian copula models with ARMA dependence and arbitrary discrete marginal distributions, provided the marginal CDF bounds are given for each observation.

Value

Returns a numeric value representing the approximate log-likelihood.

Examples

```
# Simulate Poisson AR(1) data
mu=10
tau=0.2
arma_order=c(1,0)
sim_data <- sim_poisson(mu =mu, tau=tau, arma_order=arma_order, nsim = 1000, seed = 1)
y <- sim_data$y

# Compute latent bounds for CE method
a <- qnorm(ppois(y - 1, lambda = mu))  # lower bound
```

```
b <- qnorm(ppois(y, lambda = mu))      # upper bound

# Approximate log-likelihood with CE method
llk_ce <- pmvn_ce(lower = a, upper = b, tau = tau, od = arma_order, c = 0.5)
print(llk_ce)
```

---

pmvn_ghk *GHK Log-Likelihood Approximation*

Description

Computes the approximate log-likelihood for a count time series model using the Geweke–Hajivassiliou–Keane (GHK) simulator. This method evaluates the multivariate normal rectangle probability by sequentially sampling from truncated conditionals implied by the ARMA Gaussian copula.

Usage

```
pmvn_ghk(lower, upper, tau, od, M = 1000, QMC = TRUE, ret_llk = TRUE)
```

Arguments

`lower`	A numeric vector of lower truncation bounds.
`upper`	A numeric vector of upper truncation bounds.
`tau`	A numeric vector of ARMA dependence parameters.
`od`	Integer vector `c(p, q)` specifying the AR and MA orders.
`M`	Integer. Number of Monte Carlo or QMC samples.
`QMC`	Logical. If `TRUE`, use quasi-Monte Carlo integration.
`ret_llk`	Logical. Default is `TRUE` to return log-likelihood; otherwise, return diagnostic output.

Value

If `ret_llk = TRUE`, a numeric scalar (log-likelihood); else a list containing diagnostic statistics.

Examples

```
# Simulate Poisson AR(1) data
mu=10
tau=0.2
arma_order=c(1,0)
sim_data <- sim_poisson(mu =mu, tau=tau, arma_order=arma_order, nsim = 1000, seed = 1)
y <- sim_data$y

# Compute latent bounds for CE method
a <- qnorm(ppois(y - 1, lambda = mu))  # lower bound
b <- qnorm(ppois(y, lambda = mu))      # upper bound
```

```
# Approximate log-likelihood with CE method
llk_tmet <- pmvn_ghk(lower = a, upper = b, tau = 0.2, od = c(1,0))
print(llk_tmet)
```

`pmvn_tmet` *TMET Log-Likelihood Approximation*

Description

Computes the approximate log-likelihood for a count time series model using the *Time Series Minimax Exponential Tilting (TMET)* method. This function uses the autoregressive movin average structure of the underlying Gaussian copula to compute multivariate normal rectangle probabilities via adaptive importance sampling with an optimal tilting parameter.

Usage

```
pmvn_tmet(lower, upper, tau, od, pm = 30, M = 1000, QMC = TRUE, ret_llk = TRUE)
```

Arguments

`lower`	A numeric vector of lower truncation bounds.
`upper`	A numeric vector of upper truncation bounds.
`tau`	A numeric vector of ARMA dependence parameters.
`od`	Integer vector `c(p, q)` specifying the AR and MA orders.
`pm`	Integer. Number of past lags used for approximating ARMA(p,q) to AR (required if `q > 0`).
`M`	Integer. Number of Monte Carlo or QMC samples.
`QMC`	Logical. If `TRUE`, use quasi-Monte Carlo integration.
`ret_llk`	Logical. Default is `TRUE` to return log-likelihood; otherwise, return diagnostic output.

Details

The implementation combines the Innovations Algorithm for exact conditional mean and variance computation with exponential tilting to estimate the multivariate normal probability over rectangular truncation sets.

Value

If `ret_llk = TRUE`, a numeric scalar (log-likelihood); else a list containing diagnostic statistics.

Examples

```
# Simulate Poisson AR(1) data
sim_data <- sim_poisson(mu =10, tau=0.2, arma_order=c(1,0), nsim = 1000, seed = 1)
mu=10
tau=0.2
arma_order=c(1,0)
sim_data <- sim_poisson(mu =mu, tau=tau, arma_order=arma_order, nsim = 1000, seed = 1)
y <- sim_data$y

# Compute latent bounds for CE method
a <- qnorm(ppois(y - 1, lambda = mu))  # lower bound
b <- qnorm(ppois(y, lambda = mu))      # upper bound

# Approximate log-likelihood with CE method
llk_tmet <- pmvn_tmet(lower = a, upper = b, tau = 0.2, od = c(1,0))
print(llk_tmet)
```

`predict.gctsc` *Predictive Distribution and Scoring for Gaussian Copula Time Series Models*

Description

Computes the one-step-ahead predictive distribution for a fitted Gaussian copula time series model, including summary statistics (mean, median, mode, variance) and optional scoring rules (CRPS and LOGS) if an observed value is supplied.

Usage

```
## S3 method for class 'gctsc'
predict(object, ..., method = "GHK",
                        y_obs = NULL, X_test = NULL)
```

Arguments

`object`	A fitted model object of class `gctsc`.
`...`	Not used. Included for S3 method consistency.
`method`	Character string specifying the prediction method: `"TMET"` or `"GHK"` (default: `"GHK"`).
`y_obs`	Optional observed value used to compute CRPS and LOGS.
`X_test`	Optional covariate information for prediction. Can be a named numeric vector, or a 1-row matrix/data.frame with column names matching the model covariates.

Value

`mean`	Predictive mean of Y_{t+1}.
`median`	Predictive median of Y_{t+1}.
`mode`	Predictive mode of Y_{t+1}.
`variance`	Predictive variance of Y_{t+1}.
`CRPS`	Continuous Ranked Probability Score (if `y_obs` is provided).
`LOGS`	Logarithmic score (if `y_obs` is provided).
`p_y`	Predictive pmf over $(0, \ldots, y_{\max})$.
`lower`, `upper`	95% predictive interval bounds.

See Also

`gctsc`, `arma.cormat`

`print.gctsc` *Print a gctsc Model Object*

Description

Displays the call, estimation method, parameter estimates, and likelihood information.

Usage

```
## S3 method for class 'gctsc'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	An object of class `gctsc`.
`digits`	Number of significant digits to display.
`...`	Ignored. Included for S3 method compatibility.

Value

Returns the object `x` invisibly. Called for its side effect of printing a formatted summary of the model fit.

print.summary.gctsc *Print Summary of a gctsc Model*

Description

Displays summary statistics and model fit information for a fitted gctsc model.

Usage

```
## S3 method for class 'summary.gctsc'
print(x, digits = 4, ...)
```

Arguments

x	An object of class summary.gctsc.
digits	Number of significant digits to display.
...	Ignored. Included for S3 method compatibility.

Value

Returns the input object x invisibly. Called for its side effect of printing summary information.

residuals.gctsc *Compute Randomized Quantile Residuals for Gaussian Copula Time Series*

Description

Computes residuals for a fitted gctsc object using randomized quantile residuals.

Usage

```
## S3 method for class 'gctsc'
residuals(object, method = NULL, ...)
```

Arguments

object	A fitted object of class gctsc, produced by gctsc.
method	Can be TMET or GHK
...	Ignored. Included for S3 method compatibility.

Value

A list containing:

residuals	A numeric vector of randomized quantile residuals.
pit	A numeric vector of PIT values.

References

Dunn, P. K. and Smyth, G. K. (1996), Randomized quantile residuals, *Journal of Computational and Graphical Statistics*, **5**(3): 236-244.

Examples

```
y <- sim_poisson(mu = 5, tau = 0.7, arma_order = c(1, 0), nsim = 100)
fit <- gctsc(y ~ 1, marginal = poisson.marg(), cormat = arma.cormat(1, 0),
             method = "GHK", options = gctsc.opts(seed = 1, M = 1000))
res <- residuals(fit)
hist(res$residuals, main = "Randomized Quantile Residuals", xlab = "Residual")
hist(res$pit, main = "PIT Histogram", xlab = "PIT values")
```

`sim_gctsc` *Simulate from Gaussian Copula Time Series Models*

Description

These functions simulate time series data from Gaussian copula models with various discrete marginals and an ARMA dependence structure.

Usage

```
sim_poisson(mu, tau, arma_order, nsim = 100, seed = NULL)

sim_negbin(mu, dispersion, tau, arma_order, nsim = 100, seed = NULL)

sim_zip(mu, pi0, tau, arma_order, nsim = 100, seed = NULL)

sim_binom(prob, size, tau, arma_order, nsim = 100, seed = NULL)

sim_bbinom(prob, rho, size, tau, arma_order, nsim = 100, seed = NULL)

sim_zib(prob, pi0, size, tau, arma_order, nsim = 100, seed = NULL)

sim_zibb(prob, rho, pi0, size, tau, arma_order, nsim = 100, seed = NULL)
```

Arguments

`mu`	Mean parameter(s): • For Poisson, ZIP, and negative binomial: $\mu > 0$. • Scalar or vector of length `nsim`.
`tau`	Numeric vector of ARMA coefficients, ordered as `c(phi_1, ..., phi_p, theta_1, ..., theta_q)`. ARMA(0,0) is not supported.
`arma_order`	Integer vector `c(p, q)` giving AR and MA orders.

`nsim`	Positive integer; number of time points to simulate.
`seed`	Optional integer to set the random seed.
`dispersion`	Overdispersion parameter for negative binomial marginals ($\kappa > 0$ in $\mathrm{Var}(Y) = \mu + \kappa\mu^2$).
`pi0`	Zero-inflation probability for ZIP, ZIB, and ZIBB marginals: $0 \leq \pi_0 < 1$, scalar or length `nsim`.
`prob`	Probability parameter(s) for binomial-type marginals: $0 < p < 1$, scalar or length `nsim`.
`size`	Number of trials for binomial-type marginals; positive integer (scalar).
`rho`	Intra-cluster correlation for beta-binomial and ZIBB marginals ($0 < \rho < 1$ in $\mathrm{Var}(Y) = np(1-p)[1 + (n-1)\rho]$) where n is the number of trials.

Details

Marginal types:

- Poisson: Counts with mean μ.
- Negative binomial (NB): Overdispersed counts with mean μ and dispersion parameter κ.
- Binomial: Number of successes in n trials with success probability p.
- Beta—binomial (BB): Binomial with success probability p following a beta distribution, allowing intra-cluster correlation ρ.
- Zero–inflated Poisson (ZIP): Poisson with extra probability π_0 of an excess zero.
- Zero–inflated binomial (ZIB): Binomial with extra probability π_0 of an excess zero.
- Zero–inflated beta–binomial (ZIBB): Beta–binomial with extra probability π_0 of an excess zero.

Parameterization notes:

- Negative binomial uses `dispersion` (κ) to model overdispersion: larger `dispersion` increases variance for a fixed mean.
- Beta–binomial and ZIBB use `rho` as the overdispersion parameter: ρ is the intra-class correlation, with $\rho \to 0$ giving the binomial model.
- Zero–inflated marginals include a separate `pi0` parameter that controls the extra probability mass at zero.

Value

A list with components:

- `y`: Simulated time series data.
- `z`: Latent Gaussian process values.
- `marginal`: Marginal distribution name.
- `parameters`: List of parameters used.
- `cormat`: ARMA structure.

Examples

```
# Poisson example
sim_poisson(mu = 10, tau = c(0.2, 0.2), arma_order = c(1, 1), nsim = 100, seed = 42)

# Negative Binomial example
sim_negbin(mu = 10, dispersion = 2, tau = c(0.5, 0.5), arma_order = c(1, 1))

# Beta-Binomial example with seasonal covariates
n <- 100
xi <- numeric(n)
zeta <- rnorm(n)
for (j in 3:n) {
  xi[j] <- 0.6 * xi[j - 1] - 0.4 * xi[j - 2] + zeta[j]
}
prob <- plogis(0.2 + 0.3 * sin(2 * pi * (1:n) / 12) +
             0.5 * cos(2 * pi * (1:n) / 12) + 0.3 * xi)
sim_zibb(prob, rho = 1/6, pi0 = 0.2, size = 24, tau = 0.5, arma_order = c(1, 0))
```

`summary.gctsc` *Summarize a gctsc Model Fit*

Description

Computes standard errors, z-values, and p-values for the estimated parameters in a fitted `gctsc` object.

Usage

```
## S3 method for class 'gctsc'
summary(object, ...)
```

Arguments

`object`	An object of class `gctsc`.
`...`	Ignored. Included for S3 method compatibility.

Value

A list of class `"summary.gctsc"` containing:

`call`	The model call.
`convergence`	Convergence code from the optimizer.
`coefficients`	A list with two matrices: `marginal` and `copula`, each containing estimates, standard errors, z-values, and p-values.
`loglik`	Approximate log-likelihood value.
`aic`	Akaike Information Criterion.
`bic`	Bayesian Information Criterion.

Index